Clock Group Constraints Demo Script

Introduction

This demonstration script provides high-level instructions on how to apply clock group constraints to a design using the AMD Vivado™ Design Suite.
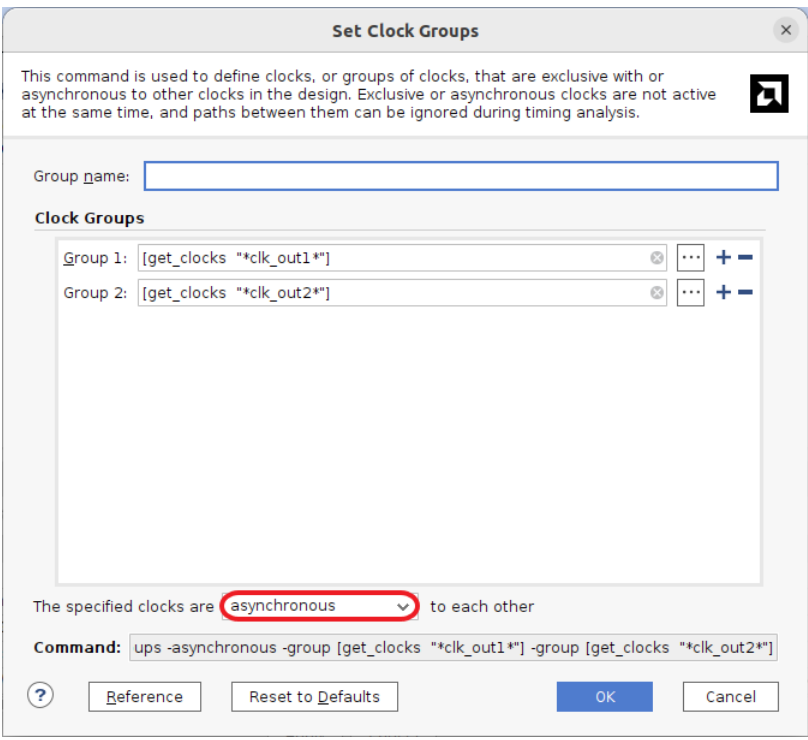
Preparation:

- Required files: `$TRAINING_PATH/Clk_Group_Constr/demo/KCU105/verilog`

- Required hardware: None

- Supporting materials: None

Clock Group Constraints

Action with Description	Point of Emphasis and Key Takeaway
- Launch the Vivado Design Suite. - Unzip the project using the Tcl Console: `exec unzip $::env(TRAINING_PATH)/ Clk_Group_Constr/demo/KCU105/ verilog.zip -d $::env(TRAINING_PATH)/ Clk_Group_Constr/demo/KCU105/ verilog`	
Open the **wave_gen.xpr** design located from the following directory: `$TRAINING_PATH/Clk_Group_ Constr/demo/KCU105/verilog`	
Open the synthesized design.	Timing is not being met in the provided design. This is because of asynchronous clock groups.
Click **Report Clock Interaction** to generate the Clock Interaction report.	The Clock Interaction report analyzes the timing paths that cross one clock domain into another clock domain. When clock frequencies are not integral multiples, such inter-clock domain paths may have unrealistic and stringent timing requirements. Such inter-clock paths are generally designed as asynchronous paths.

Action with Description	Point of Emphasis and Key Takeaway
• Analyze the clock interaction.	The display shows a grid of clock interaction. Each cell in the grid corresponds to a specific source clock (listed at the left of the grid) and a specific destination clock (listed at the top of the grid). Each cell is colored to indicate the timing/constraint status of that inter-clock path.
• Review the path between clk_out1_clk_core and clk_out2_clk_core.	When the clock frequencies are not integral multiples, such paths are designed as if asynchronous, in which case they can be marked as asynchronous clock groups (or marked as false paths).
• Enter `join [get_timing_paths -from [get_clocks clk_out1_clk_core] -to [get_clocks clk_out2_clk_core] -max_paths 200]` in the Tcl console. • Enter `join [get_timing_paths -from [get_clocks clk_out2_clk_core] -to [get_clocks clk_out1_clk_core] -max_paths 200]` in the Tcl console. • Close the Clock Interaction report.	This first command lists the paths from clk_out1_clk_core to clk_out2_clk_core. The second command will not return any paths as there are no paths from clk_out2_clk_core to clk_out1_clk_core. The list helps you to identify the paths and ascertain that these paths can safely be marked as asynchronous.
• Select **Edit Timing Constraints** from **Synthesis** > **Open Synthesized Design** to launch the Timing Constraints Editor and specify the asynchronous clock groups.	The Timing Constraints Editor enables you to create timing constraints. It analyzes the design for missing timing constraints.
• Double-click **Set Clock Groups(0)**. • Click on **Browse** next to Group1.	You can specify asynchronous clock group constraints in this category.
• Enter the search option as **NAME CONTAINS *clk_out1*** in the Group1 field. • Click **Find** and select **clk_out1_clk_core** by clicking ➡ to add and click **Set.**	To apply the asynchronous clock group constraint, you need to specify another clock group.
• Add another group of clocks by clicking the ➕ icon. • Repeat the same steps to search for ***clk_out2*** in the Group2 field and add **clk_out2_clk_core**. • Click **Set**.	Selecting the **asynchronous** option is necessary to specify this constraint.

AMD
together we advance_

Action with Description	Point of Emphasis and Key Takeaway
Set Clock Groups ⊠ This command is used to define clocks, or groups of clocks, that are exclusive with or asynchronous to other clocks in the design. Exclusive or asynchronous clocks are not active at the same time, and paths between them can be ignored during timing analysis. Group name: [ ] **Clock Groups** Group 1: [get_clocks "*clk_out1*"] ⊘ ··· + − Group 2: [get_clocks "*clk_out2*"] ⊘ ··· + − The specified clocks are (asynchronous ⌄) to each other **Command:** ups -asynchronous -group [get_clocks "*clk_out1*"] -group [get_clocks "*clk_out2*"] (?) Reference Reset to Defaults OK Cancel	
• Save the **wave_gen_timing.xdc** constraints file.	Now you have assigned the clock group constraint in the *wave_gen_timing.xdc* constraint file.
• Click **Report Clock Interaction** again to verify the user-ignored paths.	Check that the inter-clock paths are now displayed as user-ignored paths in blue, confirming the application of asynchronous clock groups between clk_out1_clk_core and clk_out2_clk_core. Two clocks are said to be asynchronous when it is impossible to determine the relative phase between them.
• Close the synthesized design.	
• Close the Vivado Design Suite.	

Summary

This demo analyzed clock interaction between the clocks and illustrated how to create an asynchronous clock group constraint in the design.

References:

- Supporting materials
  - *Vivado Design Suite User Guide (Using Constraints)* (UG903)
  - *Vivado Design Suite User Guide (Design Analysis and Closure Techniques)* (UG906).

AMD
together we advance_